Natural Question Generation using Transformers and Reinforcement Learning

Dipshikha Biswas, Suneel Nadipalli, Sneha B., Deepa Gupta, Amudha J
Department of Computer Science and Engineering
Amrita School of Computing, Bengaluru
Amrita Vishwa Vidyapeetham, India
dipshikha.biswas461@gmail.com, nsuneel89@gmail.com, snehabalaji74@gmail.com, g_deepa@blr.amrita.edu, j_amudha@blr.amrita.edu

***Abstract*— Natural Question Generation (NQG) is among the most popular open research problems in Natural Language Processing (NLP) alongside Neural Machine Translation, Open Domain Chatbots, etc. Among the many approaches taken up to solve this problem, neural networks have been deemed the benchmark in this particular research area. This paper aims at adopting a generator - evaluator framework in a neural network architecture to allow additional focus on the context of the content used for framing a question. The generator uses NLP architectures like transformers (T5) to generate a question given a context while the evaluator uses Reinforcement Learning (RL) to check the correctness of the generated question. The involvement of RL has improved the results (as shown in Table 2), and there is increased computational efficiency as the training is coupled with the policy of RL. This turns the problem into a reinforcement learning task and allows for the generation of a wide range of questions for the same context-answer pair. The given algorithm is tested on the benchmark dataset - SQuAD with BLEU score as the evaluation metric**

Keywords—Natural Question Generation, Transformers, Reinforcement Learning, SQuAD Dataset, BLEU Score

I. INTRODUCTION

Question Generation (QG) is an application area of NLP that is still relatively manual, owing to the wide range of domains and formats in which a question can be generated. Question Answering, a field adjacent to QG, is a field of research that has been explored and forms a basis for this problem. [1] models this system using statistical and semantic similarity measures and [2] focuses on legal documents. The work done in [3] follows an approach that combines word sense disambiguation with semantic role labelling and [4] steers this task to the medical domain.

Another domain that forms the base for this work is Paraphrase Detection, as that also relies on context. [5] explores this task in Indian Regional Languages and [6] provides an LSTM based architecture for the same.

Natural Question Generation (NQG), along with Neural Machine Translation and Open Domain Chatbots, is one of the most prominent open research challenges in NLP. The generation is how it is described and the difficulty of generating syntactically correct, semantically sound, and pertinent questions from a range of input formats, such as text, documents, a structured database, or a knowledge base, has been formally defined. Mundane tasks such as generating questions from a textbook for educational purposes could benefit from this research area as it does away with the need for human involvement. Other applications of NLG include generating FAQs for a website and maintaining the natural flow of chatbots.

Sequence-to-sequence (Seq2Seq) learning, a neural network-based technique, has recently shown exceptional performance on a range of NLP tasks, including question production. The task of generating questions from a triplet of topic, relation, and object is simplified in a novel deep learning method for NQG [7]. A Seq2Seq approach is shown in the Learning to Ask (L2A) model [8] with an emphasis on question generation from the text. [9] encoded ground-truth solutions and used bi-directional LSTMs in a Seq2Seq context. Additionally, they use context matching and the copy mechanism [10] to record interactions between the given ground-truth response and its context in the passage.

- The proposed approach focuses on a generator-evaluator framework,
  - **Generator** – The Neural Network model that aids in the generation of the question
  - **Evaluator** – The RL Framework that checks if the question is appropriate or not.
- A custom loss function is modelled by combining RL rewards along with the model loss. This function improves upon BLEU score, by considering the semantics of the question generated.
- Apart from comparing the BLEU scores of the model generated and ground truth question, this research work introduces another metric that allows the comparison of the answer generated from the model generated question and the ground truth answer.

The rewards used are not only based on the question generated by the model as [11], but also the answer generated from this question. The comparison of BLEU [11] scores is done as a part of the result as they are the standard metric.

In Section II, papers containing similar works are reviewed and discussed. The following section, Section III details the dataset used while Section IV follows a discussion on the system architecture which details the RL model, its formulation which includes the state space, action space and rewards, and the neural network, along with the general workflow. Section V entails the software/hardware setup as well as the evaluation metrics and Section VI presents the results of this work in a tabular format. Section VII and VIII conclude this work and mention the next steps for its improvement.

II. RELATED WORK

Researchers have approached natural question generation with a variety of methods, with the most popular being some variation of a generator-evaluator framework, in which the generator is used to form a question-answer pair,

2022 OITS International Conference on Information Technology (OCIT) | 978-1-6654-9348-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/OCIT56763.2022.00061

978-1-6654-9348-2/22/$31.00 ©2022 IEEE
DOI 10.1109/OCIT56763.2022.00061

which is then evaluated by the evaluator mechanism. These methods frequently also include complicated rewards that are in line with performance indicators (such as BLEU and ROUGE on test data in the evaluator). Other approaches involve monitoring word coverage in the context of QG from paragraphs using [12] a max out pointer network. Another innovative approach to this issue is to simulate how a human might ask a question [13]; to do this, a question must take into account three different elements: a response, a clue, and a style. This turns the whole project into a one-to-one problem, making learning easier. In line with this method, comes another approach, this time harnessing the power of GANs [14]. The model converts question generation into a one-to-one problem and the generative model of the GAN learns a distribution over potential questions, leading to more variety of the generated questions; while parallelly modifying the discriminator the push the generative model to produce questions that come close to questions generated by humans.

As mentioned earlier, one of the more popular approaches also involves designing custom rewards, often ones that seek to address certain research gaps in the field of NQG. One such set of rewards put forth focused on the semantics of the question being generated [15]. They found that since most QG models at that point in time were learned through strict teacher forcing rather than with the extra aid of semantic regularization, the generated questions seemed to have deviated in terms of their semantics, a problem they termed as "semantic drift". To address this, they came up with two semantic based rewards, resulting in a significant improvement in results. A rather unconventional approach to the problem looked at it from a different perspective, choosing to focus on Conversational Question Generation (CQG) [16]. The model focused on generating interconnected questions based on conversation history.

Another work looks at a two-step approach by first identifying key phrases that could constitute answers in a passage and use those as a base [17]. With this they hope to address problems such as out-of-domain words, style of linguistics among others. There are certain other works centered around Neural Networks. One such model focuses on the answer embedding, purely focusing on the answer for question generation, along with encoding the relative distance between the context words and the answer as well. This helps the model choose context words located close to the answer [18]. One paper takes an approach fundamentally equivalent to the generator-evaluator framework in that this involves an encoder-decoder mechanism. The document and answer are fed to the encoder and the decoder sequentially outputs words of the question that are conditioned on said document and answer [19]. BERT is another avenue employed for the task of NQG, as seen in [20]. This paper actually serves as a comparative study as well, employing 3 different neural architectures that are variations of BERT. Another, rather unique approach, looks at using certain criteria to pick *question-worthy* sentences from a given passage and pass that in as context to a seq2seq model with an attention mechanism [21].

There is another side of Natural Question Generation that incorporates Reinforcement Learning into the framework. One common approach involves modifying [22] the generator framework by incorporating an RL reward function into the loss function. This was done by introducing two custom RL reward functions, both providing a more lenient variant of BLEU scores, that simply looks at exact n-grams matching. The results obtained in this work were fruitful with a BLEU score of 2.09. Another work, like this, devised 3 separate reward functions, each relating to the quality of the generated question, namely answerability, fluency, and relevance [23]. A variation of this would be passing in the answer span as a part of the input and using a self-attention head along with an LSTM layer [24]. The question generator here is a seq2seq with a gated self-attention network and glove embeddings and the evaluator is based on the principle of self-critical-sequence-training (SCST), which utilizes test-time inference output to normalize reward. This involves 4 RL rewards as well: fluency, similarity, answerability, and relevance. A graph to sequence model was explored in [25] which has a hybrid evaluator with a mixed aim incorporating both cross-entropy and RL losses to ensure the generation of syntactically and semantically correct text and a bidirectional gated graph neural network-based encoder to embed the passage. Another interesting approach in the same category involves integrating and properly using answer information and focusing on increasing the depth of comprehension [26]. Both the document and answer are given a proper representation and fed to a decoder, followed by the introduction of a semantic based attention module in order to integrate information.

Some previous works focused on incorporating a self-attention mechanism, without the use of RL [27]. This, however, does not produce appropriate questions for a variety of context. Another approach was based on Contrastive Learning [28] where a question is generated from a given positive as well as negative sample of the context, where the answer must lie close to the positive sample and far away from the negative sample. An attempt at Neural Networks with answer separation was also taken [29], where their algorithm learns to determine which interrogative word should be used by swapping out the target answer in the original section for a unique token.

Overall, a few works have discussed the approach to the problem as a purely NLP problem whereas some have considered it as a combination of both NLP and RL. With just NLP, usually, the approach was to refer to the answers of the questions generated rather than focusing on the generation itself using models like LSTM, Seq2Seq and SCST. Previous work related to RL in NLP have been revolving around implementing a predefined reward function that is usually a combination of other related semantic based reward functions such as fluency, similarity etc. Though NQG works well even without the RL component, adding a reward function makes the generation accurate overall.

In this research work, two custom RL rewards namely, Question Sentence Overlap Score and Predicted and Encoded Answer Overlap score have been used to produce accurate questions as the output. BLEU [30] scores of the questions and answers are evaluated to take the context of the question into account. The benchmark SQuAD [31] dataset has been used for this work.

III. Dataset Description

The dataset used for this research paper is the SQuAD (Stanford Question Answering Dataset). SQuAD is considered as a benchmark dataset for many NLP tasks and is evaluated against as a measure for a given model's performance. This dataset contains a collection of 107,785 question-answer pairs along with their context derived from 536 Wikipedia articles in English. It amounts to 70MB of data

The questions and answers span over a wide domain of subjects. This dataset was trifurcated into training, development, and testing sets, with 70% of the articles going into the training set and the rest 20% and 10% for testing and validation respectively. All special characters are removed from this dataset and it is converted to lowercase.

IV. Proposed Working Principle

The block diagram represented in fig.1 depicts the question generation system, consisting of 4 stages. The core of the system is the generator-evaluator framework, where the input is a question-context-answer triplet. In the first stage, the triplet is passed through T5's encoder, which is then used to generate a question, again using T5. The third stage involves using the custom RL reward to evaluate this generated question against the ground truth question. The result of this evaluation is then fed back into the system that also acts as an RL agent, which uses it to improve its learning. The final stage involves decoding the generated question to get in the form of a human readable format.

The encoder shown in fig.2 is T5's inbuilt encoder. The question from the triplet is passed in as it is and T5 creates a vector representation using the vocabulary from T5. The input sentence (question) is tokenized and mapped to a sequence of embeddings. This sequence is then passed to the encoder, the layers of which are detailed in fig.2.

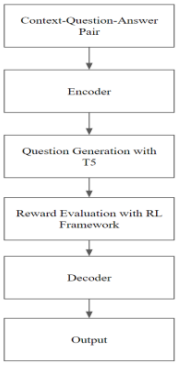

Fig 1. General Workflow

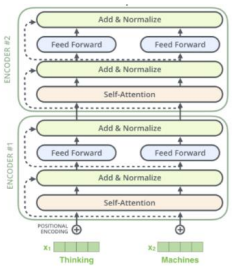

Fig 2. T5 Encoder Architecture

A. Generator Framework – Question Generation

Fig.1 contains the core of the proposed model – the generator-evaluator framework. The framework consists of two components; the generator and the evaluator. The generator's foundation is a T5 transformer, which also acts as the agent. This T5 transformer also performs the task of question generation, which accepts a triplet consisting of a context, question and an answer. Prior to training, the transformer was fine tuned for question generation by using Transfer Learning on the existing architecture with the SQuAD dataset. The input format for the model is as follows:

- **Context –** This encapsulates a sequence of words in the form of a paragraph, from which a question is asked
- **Answer –** This contains word/words from the context, that answer the question.

These parameters are encoded before being processed for the task of question generation. The final output of this generator is a question, formed using the given parameters. At both the encoder and decoder layer of this architecture, a self-attention mechanism is used to throw light on the appropriate words while framing the question.

In essence, this problem is viewed as an RL problem and this generator framework, consisting of the T5 transformer, acts as the agent.

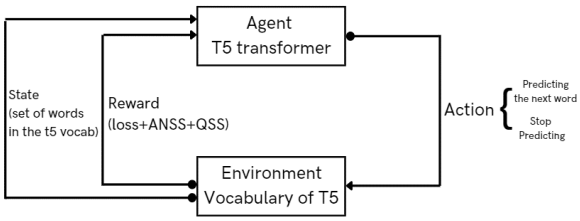

Fig 3. RL Environment

B. RL Formulation

In order to improve learning while training, the task is viewed as an RL environment, and all of the components in the system mapped to the standard components of an RL environment. This environment is depicted in fig.3. The components of the RL environment can be broken down as follows:

- **Environment**: The complete vocabulary of the transformer (T5). This is a fully observable, deterministic environment
- **States**: The set of words contained in the vocabulary of the transformer (T5). The number of states is 32100, corresponding to the number of words in T5's vocabulary.
- **Action**: This environment involves two actions. One is to predict the next word and the other is to stop predicting. Done using T5
- **Policy**: Gradient Descent is used twofold here - To minimize the loss faced by T5 and to maximize return
- **Reward**: The weighted sum of two reward functions, QSS and ANSS

- **Agent**: The generator which is the T5 Transformer in this paper
- **State Space**: This is |v| x |v|, which in this case is 32100 x 32100

Action Space: Action Space with two actions - Predicting the next word and stopping prediction.

C. Evaluator Framework – RL Reward

The latter half of the pipeline, the evaluator, evaluates the generated question, the output of the previous stage. Usually in a question generation task, the evaluation is done using BLEU, a standard metric. Though these are standard scores for text-to-text tasks, using these metrics may not necessarily be the best fit. These scores only compare the generated and target question, using n-gram matches and this fails in evaluating questions that have the same meaning but different structure.

This can be overcome by transforming the problem from a supervised task into an RL task. Incorporating the reward functions of QSS and ANSS into the model, along with utilizing Gradient Descent as the environment's policy helps the environment learn better and more diverse questions. The involved metrics are as follows:

- **QSS (Question Sentence Overlap Score):** This reward function is unique to the QG. The sentence overlap score is determined by counting how many n-grams are comparable between the source phrase and the predicted query. This incentive guarantees that the produced question is appropriate for the sentence. As a result, if precision (s, q) determines the n-gram precision match between the phrase and the query.

$$QSS = \left(\prod_{i=1}^{n} precision_i(sentence, question)\right)^{\frac{1}{n}} \quad (1)$$

- **ANSS (Predicted and Encoded Answer Overlap score):**
  To ensure that the generated question is in relation to the crucial/ground truth response, the response overlap score is computed. The answer overlap score is the ratio of similar n-grams between the encoded response and the answer predicted (ans) for the produced question using the best question-answering model over SQuAD.

$$ANSS\left(\prod_{i=1}^{n} precision_i(ans_{qa}, pivotal\_answer)\right)^{\frac{1}{n}} \quad (2)$$

- **Custom Loss Function:** This custom loss function serves as the cumulative reward for the RL agent. It is a weighted combination of the QSS, ANSS and the standard loss. The weighting factor is *gamma*, also known as the discounting factor for an RL reward. It is used to make sure none of the rewards taken into consideration are given more importance than the other. This usually takes a value between 0.1 and 0.3.

$$loss = -[(\gamma * QSS) + (\gamma * ANSS)] + output \quad (3)$$

Thus, this custom loss function, which encompasses two RL reward functions as well as the model's loss forms a base for training the model. The RL functions focus on training the model to generate questions while keeping the context in mind, while the loss of the model serves as the policy for this training, which is Gradient Descent. This acts as an optimization problem that does two tasks simultaneously – Decreases the model loss and increases the reward values. As given in the equation, we decrement the negative reward value and the loss, thus overall minimizing the value of this custom loss.

The Neural Network embedded in the T5 Transformer acts as the Q-Table for the environment. A Neural Network Q-Table is an appropriate choice for this problem due to the substantial number of states in the environment. An episode is defined as the generation on a single question, and as the environment runs through more episodes, it updates the Q-Table with better and better values and can take a better decision on when to keep generating the next word and when to stop. The final trained RL agent, i.e., the T5 Transformer, can take in a context-answer pair and generate the top K questions with the help of Beam Search.

D. Evaluation

After the training of the model using the custom loss function in (3), the generated questions are evaluated using two methods:

- **Question based BLEU Score (BLEU Score Q)** – The BLEU score between the model-generated question and the ground truth question is evaluated
- **Answer based BLEU Score (BLEU Score A)**– The BLEU score between the generated answer from the model-generated question and the ground truth answer is calculated.

V. Experimental Setup

This research work was implemented in Python, and was hosted on the Google Colaboratory virtual environment. The environment was run in GPU mode, where allocated memory was 12GB of RAM and 1 CPU Core was utilized. For the evaluation of Answer based BLEU Score, the question that is generated is passed through Hugging Face's Roberta Model. In this scenario, the Roberta model acts as a Question-Answering system, where the context and the generated question are passed as an input and answer to that question is provided as the output. The default hyperparameters are considered for the T5 model.

The hyperparameters for the RL model have been enumerated in Table 1.

S No.	Hyperparameter	Value
1	Number of Epochs	10
2	Optimizer	Adam
3	Batch Size	4
4	Sample Size	107,785
5	Gamma (discount factor)	0.3
6	Alpha (learning rate)	3×10^{-4}
7	Epsilon (stopping criteria)	10^{-8}

Table 1. Hyperparameter values for the T5+RL model

VI. Experimental Results and Analysis

The tables are a consolidation of all the experiments performed in this research work. As was to be expected, the method that involves the combination of T5 and RL shows the best performance. This could be due to the custom loss function that takes into consideration the context and comes up with the best question for the same.

Table 2 displays a comparison of this work's results with the reference paper against standard metrics. Table 3 displays a comparison of the models used in this paper against custom metrics, followed by an explanation for the same. And finally, Table 4 showcases and handful of test cases after being run through the trained T5+RL agent. The first two rows display two cases wherein the predicted questions are a rephrasing of the existing question, with just a few words added/removed. As a result of this, both predicted and actual answers are the exact same, leading to a BLEU score of 1 as well.

The last two rows are examples of question generation deviates from the ideal path and leads to a lower BLEU score. The fourth pair of questions are synonymous with each other. But because of the slightly different structure in the questions, the answers are slightly different, explaining the average BLEU score of 0.7. The last row is an example of a less than average generation done by the model, where the question-answer pairs are an inverse of the other – the ground truth question asks for a definition and the ground truth answer answers it, whereas the generated question contains the definition of the term and the generated answers produces that same term. This results in a low albeit appropriate BLEU score of 0.58. These results showcase the proposed model's different approaches to generating questions by taking into account the relevant words and the structure of the given question, along with some of its imperfections, showing the definite scope for improvement.

The plot depicts the comparison of BLEU scores against the RL reward for each sample in the testing dataset. It was found that the reward fluctuates in the beginning due to the random steps taken by the baseline agent. As the episodes progress, the reward stabilizes and shows a much better performance than the BLEU score.

S No	Model	BLEU-Q	BLEU-A	Custom Loss
1	BiLSTM + RL [11]	2.05	-	-
2	BiGRU + RL [23]	1.71	-	-
3	T5	1.99	0.7	0.5
4	***T5+RL (proposed model)***	***2.69***	***0.89***	***0.07***

Table 2. *Comparison of models against BLEU score*

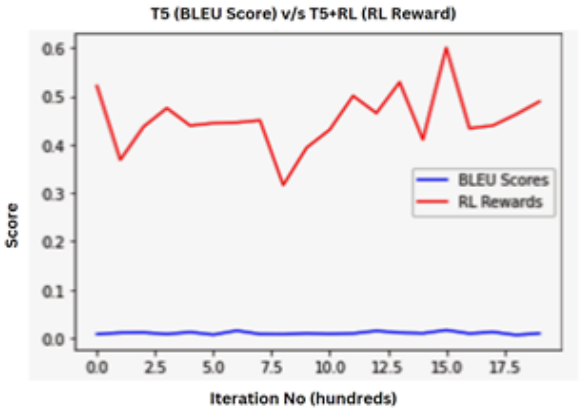

Fig 3. *Plot of RL Rewards v/s BLEU Score*

VII. Conclusion and Future Scope

The generator in this project is the RL agent, and at each time step in the episode, it either generates a word or stops generating it. Each episode is defined as the generation of one question. The evaluator, which incorporates the RL reward function and the policy, evaluates the generated question and helps the model learn. The agents used in the comparative study are both pre-trained transformers from Hugging Face, T5 and BART. The experiments performed proved fruitful with exceptional results. The RL reward towers over the NLP metrics, proving the strength of the integration of RL concepts into a task conventionally classified as an NLP application.

Future improvements include performing Named Entity Recognition (NER) and Parts of Speech (POS) tagging for ease of relevance and formulating different types of questions such as MCQs and True or False questions.

S No	Actual Question	Predicted Question	Actual Answer	Predicted Answer	BLEU-A
1	What Islamic denomination was Avicenna thought to be a member of?	What is the background of Avicenna's family?	Sunni	Sunni	1.0
2	What was the name of the airport the United States built on Ascension Island?	What airport was built during the Second World War?	Wideawake airport	Wideawake airport	1.0
3	What Prime Minister objected to the name change?	Who favored the retention of the House of Windsor?	Winston Churchill	The British Prime Minister, Winston Churchill, and Elizabeth's grandmother, Queen Mary	0.7
4	In what classification is a person who gives up their exclusive sense of self and defines him or herself only in terms of social engagement?	What is the relational self?	the relational self	a perspective by which persons abandon all sense of exclusive self	0.58

Table 3. *Sample generated question from the T5+RL model*

REFERENCES

[1] Dwivedi, G., Venugopalan, M., Gupta, D. (2020). A Statistical-Semantic PSO Model for Customer Reviews-Based Question Answering Systems. In: Reddy, V., Prasad, V., Wang, J., Reddy, K. (eds) Soft Computing and Signal Processing. ICSCSP 2019. Advances in Intelligent Systems and Computing, vol 1118. Springer, Singapore. https://doi.org/10.1007/978-981-15-2475-2_13

[2] G, Veena, Deepa Gupta, Akshay Anil and Akhil M S. "An Ontology Driven Question Answering System for Legal Documents." 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT) 1 (2019): 947-951.

[3] L. R. Pillai, Veena, G., and Gupta, D., "A Combined Approach Using Semantic Role Labelling and Word Sense Disambiguation for Question Generation and Answer Extraction", in Second International Conference on Advances in Electronics, Computers and Communications (ICAECC), Bangalore, India, 2018

[4] R. S. Nambiar and D. Gupta, "Dedicated Farm-Haystack Question Answering System for Pregnant Women and Neonates Using Corona Virus Literature," 2022 12th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2022, pp. 222-227, doi: 10.1109/Confluence52989.2022.9734125.

[5] Anand Kumar, M., Singh, S., Kavirajan, B., Soman, K.P. (2018). Shared Task on Detecting Paraphrases in Indian Languages (DPIL): An Overview. In: Majumder, P., Mitra, M., Mehta, P., Sankhavara, J. (eds) Text Processing. FIRE 2016. Lecture Notes in Computer Science, vol 10478. Springer, Cham. https://doi.org/10.1007/978-3-319-73606-8_10

[6] A. D. Reddy, M. Kumar, A., Dr. Soman K. P., G.R., M. Reddy, V.S., R., and V.K., P., "LSTM based paraphrase identification using combined word embedding features", in Advances in Intelligent Systems and Computing, 2019, vol. 898, pp. 385-394.

[7] Iulian Vlad Serban, Alberto Garc´ıa-Duran, Caglar ´ Gulcehre, Sungjin Ahn, Sarath Chandar, Aaron Courville, and Yoshua Bengio. 2016. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus

[8] Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension

[9] Linfeng Song, Zhiguo Wang, Wael Hamza, Yue Zhang, and Daniel Gildea. 2018. Leveraging context information for natural question generation

[10] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer generator networks

[11] Kumar, Vishwajeet & Ramakrishnan, Ganesh & Li, Yuan-Fang. (2019). Putting the Horse before the Cart: A Generator-Evaluator Framework for Question Generation from Text. 812-821. 10.18653/v1/K19-1076.

[12] Yao Zhao, Xiaochuan Ni, Yuanyuan Ding, and Qifa Ke. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks

[13] Bang Liu, Haojie Wei, Di Niu, Haolan Chen, Yancheng He. 2020. Asking Questions the Human Way: Scalable Question-Answer Generation from Text Corpus

[14] Yao, Kaichun & Zhang, Libo & Luo, Tiejian & Tao, Lili & Wu, Yanjun. (2018). Teaching Machines to Ask Questions. 4546-4552. 10.24963/ijcai.2018/632

[15] Zhang, Shiyue & Bansal, Mohit. (2019). Addressing Semantic Drift in Question Generation for Semi-Supervised Question Answering.

[16] Gao, Yifan & Li, Piji & King, Irwin & Lyu, Michael. (2019). Interconnected Question Generation with Coreference Alignment and Conversation Flow Modeling.

[17] Subramanian, Sandeep & Wang, Tong & Yuan, Xingdi & Trischler, Adam. (2017). Neural Models for Key Phrase Detection and Question Generation.

[18] Sun, Xingwu & Lyu, Yajuan & He, Wei & Ma, Yanjun & Wang, Shi. (2018). Answer-focused and Position-aware Neural Question Generation. 3930-3939. 10.18653/v1/D18-1427.

[19] Yuan, Xingdi & Wang, Tong & Gulcehre, Caglar & Sordoni, Alessandro & Bachman, Philip & Zhang, Saizheng & Subramanian, Sandeep & Trischler, Adam. (2017). Machine Comprehension by Text-to-Text Neural Question Generation. 15-25. 10.18653/v1/W17-2603.

[20] Chan, Ying-Hong & Fan, Yao-Chung. (2019). A Recurrent BERT-based Model for Question Generation. 154-162. 10.18653/v1/D19-5821.

[21] Sedigheh Mahdavi, Aijun An, Heidar Davoudi, Marjan Delpisheh, and Emad Gohari. 2020. Question-Worthy Sentence Selection for Question Generation. In Advances in Artificial Intelligence: 33rd Canadian Conference on Artificial Intelligence, Canadian AI 2020, Ottawa, ON, Canada, May 13–15, 2020, Proceedings. Springer-Verlag, Berlin, Heidelberg, 388–400

[22] Vishwajeet Kumar, Kireeti Boorla, Yogesh Meena, Ganesh Ramakrishnan, and Yuan-Fang Li. 2018. Automating reading comprehension by generating question and answer pairs

[23] Yuxi Xie, Liangming Pan, Dongzhe Wang, Min-Yen Kan, Yansong Feng. 2020. Exploring Question-Specific Rewards for Generating Deep Questions

[24] Peide Zhu, Claudia Hauf. 2021. Evaluating BERT-based Rewards for Question Generation with Reinforcement Learning

[25] Chen, Yu & Wu, Lingfei & Zaki, Mohammed. (2019). Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation.

[26] Wang, Liuyin & Xu, Zihan & Lin, Zibo & Zheng, Haitao & Shen, Ying. (2020). Answer-driven Deep Question Generation based on Reinforcement Learning. 5159-5170. 10.18653/v1/2020.coling-main.452.

[27] Du, Xinya & Shao, Junru & Cardie, Claire. (2017). Learning to Ask: Neural Question Generation for Reading Comprehension. 1342-1352. 10.18653/v1/P17-1123

[28] Cho, Woon & Zhang, Yizhe & Rao, Sudha & Celikyilmaz, Asli & Xiong, Chenyan & Gao, Jianfeng & Wang, Mengdi & Dolan, Bill. (2021). Contrastive Multi-document Question Generation. 12-30. 10.18653/v1/2021.eacl-main.2.

[29] Kim, Yanghoon & Lee, Hwanhee & Shin, Joongbo & Jung, Kyomin. (2019). Improving Neural Question Generation Using Answer Separation. Proceedings of the AAAI Conference on Artificial Intelligence. 33. 6602-6609. 10.1609/aaai.v33i01.33016602.

[30] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation.

[31] Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). SQuAD: 100,000+ Questions for Machine Comprehension of Text. arXiv. https://doi.org/10.48550/arXiv.1606.0

.